Marist College
MS in Computer Science
School of Computer Science and Mathematics

CMPT-435L-111-23s Algorithm Analysis and Design
Brian Gormanly
Spring 2023

Assignment 0: Git and LaTeX

Connor H. Johnson
connor.johnson1@marist.edu

Assignment 0: Git and LaTeX

Connor H. Johnson
connor.johnson1@marist.edu

January 26, 2023

Overview

Within this document, I would like to go step-by-step to give you an overview of what my favorite piece of code does. Off of personal preference, I enjoy doing a lot of front-end development. Seeing what I produce brings me back to languages like HTML, EJS, CSS, and many more. I find that running my websites off of localhost and using Nodemon is the easiest way to produce my work quickly and for me not to go through continuous tedious steps to host my website. This document will show the mandatory code and package installations to host a localhost web page using Nodemon properly for files in EJS and CSS.

Step-by-Step Instructions

To begin, you will need to start a new file inside your code editor. My favorite is Visual Studio Code (VSCode), as I find the environment very clean and easy to comprehend, unlike other editors. For more information about VSCode, I recommend visiting their website[1]

1 Installation of the Packages:

Once you obtain a blank document inside VSCode, we will open a new terminal.

To open the terminal:

- Use the `Windows: Ctrl+, Linux: Ctrl+`` keyboard shortcut to toggle the terminal panel.
- Use the `Windows: Ctrl+Shift+, Linux: Ctrl+Shift+`` keyboard shortcut to create a new terminal.
- Use the **View > Terminal or Terminal > New Terminal** menu commands.
- From the Command Palette `Windows: Ctrl+Shift+P, Linux: Ctrl+Shift+P` , use the View: Toggle Terminal command.

[1]Visual Studio Code: https://code.visualstudio.com/

Now that the terminal is open, we are going to have to use the 'npm' command to install our packages into our project. Once the code provided below is executed into the terminal, we now have installed a `node_modeules`, `package-lock.json`, and a `package.json` file so that our website can host properly.

Package installation commands:

1. `npm init -y`
2. `npm i express ejs`
3. `npm i -D nodemon`

2 Edit the Package

Although we installed these packages, there are still some changes we will have do for them to work properly.

Below is what your `package.json` file should look like:

```
{
  "name": "johnsonwebsite",
  "version": "1.0.0",
  "description": "My personal website -- soon to be hosted -- currently ran by loca
  "main": "index.js",
  "scripts": {
    "scripts":{"start":"echo \ "Error: no test specified:\""&& exit 1 "
},
  },
  "repository": {
    "type": "git",
    "url": "git+https://github.com/connorjohnson6/JohnsonWebsite.git"
  },
  "keywords": [],
  "author": "",
  "license": "ISC",
  "bugs": {
    "url": "https://github.com/connorjohnson6/JohnsonWebsite/issues"
  },
  "homepage": "https://github.com/connorjohnson6/JohnsonWebsite#readme",
  "dependencies": {
    "ejs": "^3.1.8",
    "express": "^4.18.2",
    "express-ejs-layouts": "^2.5.1"
  },
  "devDependencies": {
    "nodemon": "^2.0.20"
  }
}
```

However, we will change the ”main”(line 5) and add some functions to our ”scripts”(Line 7). We are doing this mainly to communicate with our soon-to-be-created app.js file and to add some shortcuts to properly call our Nodemon package in our terminal for our web page to run when no errors occur continuously.

Below is what your package.json file should look now look like when edited:

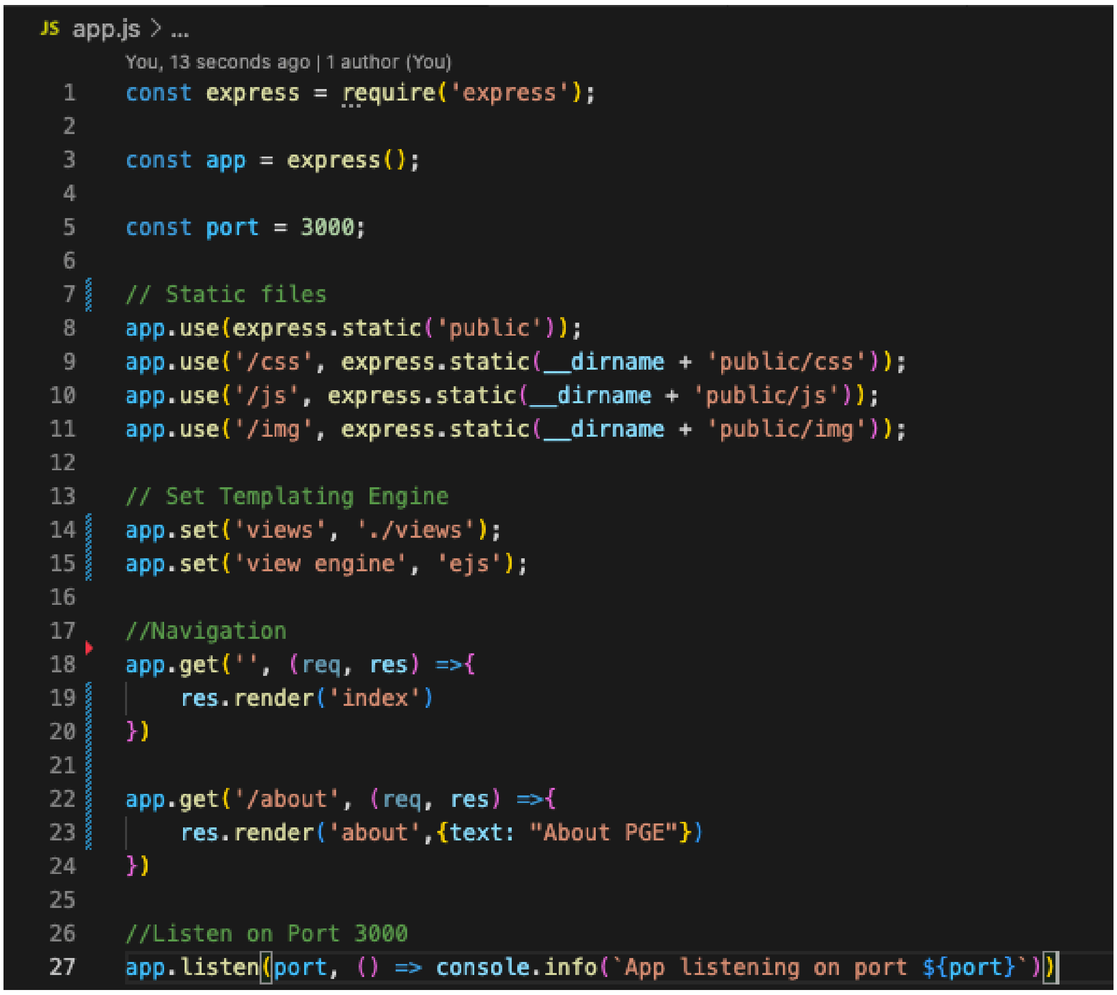
```
{
  "name": "johnsonwebsite",
  "version": "1.0.0",
  "description": "My personal website -- soon to be hosted -- currently ran by loca
  "main": "app.js",
  "scripts": {
    "start": "node app.js",
    "dev": "nodemon app.js"
  },}}
  "repository": {
    "type": "git",
    "url": "git+https://github.com/connorjohnson6/JohnsonWebsite.git"
  },
  "keywords": [],
  "author": "",
  "license": "ISC",
  "bugs": {
    "url": "https://github.com/connorjohnson6/JohnsonWebsite/issues"
  },
  "homepage": "https://github.com/connorjohnson6/JohnsonWebsite#readme",
  "dependencies": {
    "ejs": "^3.1.8",
    "express": "^4.18.2",
    "express-ejs-layouts": "^2.5.1"
  },
  "devDependencies": {
    "nodemon": "^2.0.20"
  }
}
```

3 Add your folders and files

Now that we properly have everything downloaded, we will make a file named `app.js`(1) and two folders named **views**(2) and **public**(3).

(1) App.js

- The code below is mandatory as this is how you implement the Express and EJS packages. Line 1-3 allows you to use express while the next line assigns a port number you want to run your LocalHost from. Many choose their port number to be, for example, 3000, 4000, or 5000.
- Following that are the static files that allow your soon-to-be-created EJS files to connect to each other and allow you to use CSS and JavaScript and upload images.
- In your next section of code, you're to set up your engine. On line 14, you can see that we have our views folder named in the **app.set** as this is what we will call from due to it soon holding our EJS files.
- The rest of your code will be your navigation within your search bar. These six lines of code are what you will put into your navbar when your host your website. For example, `Localhost:3000/about` will bring you to your **about page** later created in your views folder. Following that is an `app.listen` command that will communicate with express to host your page via your desired port.



```
1  const express = require('express');
2
3  const app = express();
4
5  const port = 3000;
6
7  // Static files
8  app.use(express.static('public'));
9  app.use('/css', express.static(__dirname + 'public/css'));
10 app.use('/js', express.static(__dirname + 'public/js'));
11 app.use('/img', express.static(__dirname + 'public/img'));
12
13 // Set Templating Engine
14 app.set('views', './views');
15 app.set('view engine', 'ejs');
16
17 //Navigation
18 app.get('', (req, res) =>{
19     res.render('index')
20 })
21
22 app.get('/about', (req, res) =>{
23     res.render('about',{text: "About PGE"})
24 })
25
26 //Listen on Port 3000
27 app.listen(port, () => console.info(`App listening on port ${port}`))
```

(2) **Views**

- Now that we have our express working and hosting a live port using Nodemon, we can customize our views folder. Inside of here will be our HTML, or in this case, EJS files. We will be creating two EJS files, **index.ejs** and **about.ejs**. Just so nothing gets complicated, inside our index.ejs and about.ejs, we can insert a `!` tag and press enter to insert the needed parameters automatically.
  The HTML code will look like this:

```
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta http-equiv="X-UA-Compatible" content="IE=edge">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>Document</title>
</head>
<body>

</body>
</html>
```

  Once this is done, we must link our CSS page inside our index.ejs file. In addition, we will also add a title to our page and a `<h1>Hello World<\h1>` so that we know that the website is working.
  The HTML code will now look like this:

```
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta http-equiv="X-UA-Compatible" content="IE=edge">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <link rel="stylesheet" href="/style.css">
    <title>Index</title>
</head>
<body>

    <h1>Hello World</h1>

</body>
</html>
```

(3) **Public**

- Now that we have already planned for a CSS page inside of our index.ejs page, we now must add a file called `style.css` inside of our public folder
- since a CSS folder will be filled with your own code to customize, for now, we are just going to make the background blue just so we can see if the CSS is correctly implemented inside of our index.ejs page.
  The CSS code we will include is:

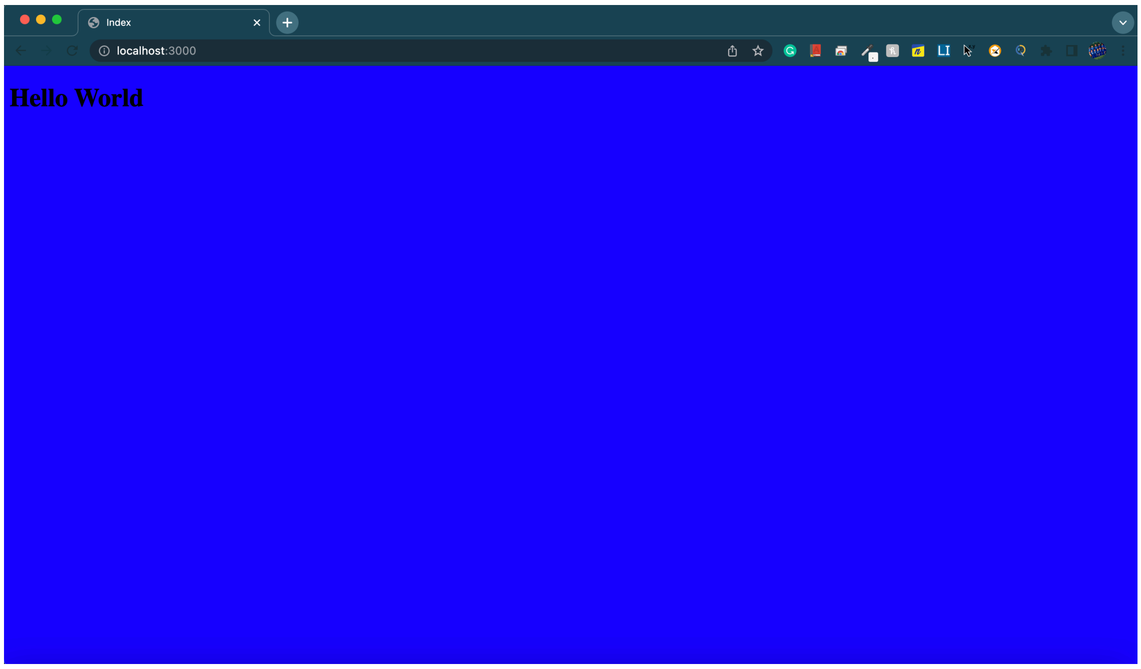
```
body{
    background-color: blue
}
```

4 Live hosting your website

Now that we have everything properly set up, we will now host our website on a browser using a local host
Inside our terminal in VSCode, we will now write and enter:

- `npm run dev`

When we enter `localhost:3000` in the browser of your choice, the result should be:



Hello World

5 Customization

Now that you have successfully hosted a server and connected your EJS and CSS files, you can now start customizing your website to your liking!